PR #27545 完整报告

sgl-project/sglang

Fix NaN in triton EAGLE spec-v2 draft-extend CUDA graph at topk>1 (wrong qo_indptr stride)

合并时间：2026-06-08 17:28

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/27545

执行摘要

- 一句话：修复 EAGLE spec-v2 draft-extend CUDA Graph 中 qo_indptr stride 错误
- 推荐动作：值得精读。该 PR 展示了一种典型的 CUDA Graph 元数据布局 Bug：indptr stride 与运行时 token 布局不一致，且容易漏掉 downstream 的 max_extend_len 同步修复。对于维护 spec-v2 或其他 CUDA Graph 元数据构建的开发者有参考价值。

功能与动机

当 topk>1 时，CUDA Graph 模式下 draft-extend 阶段产生 NaN。PR body 明确指出：draft-extend CUDA-graph metadata 构建的 qo_indptr 使用 num_steps+1 作为 stride，但 runner 实际每请求布局 num_draft_tokens 个 token，两者在 topk>1 时不一致，导致 bs>1 时读错偏移。

实现拆解

1. 修复 qo_indptr stride（_update_draft_extend_buffers）：将原硬编码的 self.speculative_num_steps + 1 改为按 forward_mode 条件选择：V2 draft-extend 时用 self.num_draft_tokens，否则保持原值。
2. 修复 max_extend_len（_build_cuda_graph_forward_metadata）：对应地将 draft-extend 分支中的 max_extend_len 也从固定值改为条件选择，确保 Triton extend kernel 的计算网格与 token 布局一致。
3. 两处改动均添加了明确注释解释原因，提升可维护性。

关键文件：

- python/sglang/srt/layers/attention/triton_backend.py（模块 注意力层；类别 source；类型 core-logic；符号 _update_draft_extend_buffers, _build_cuda_graph_forward_metadata）：唯一修改文件，包含两处核心修复：qo_indptr stride 和 max_extend_len 的条件选择，直接决定了 CUDA Graph 元数据的正确性。

关键符号：_update_draft_extend_buffers, _build_cuda_graph_forward_metadata

关键源码片段

python/sglang/srt/layers/attention/triton_backend.py

唯一修改文件，包含两处核心修复：qo_indptr stride 和 max_extend_len 的条件选择，直接决定了 CUDA Graph 元数据的正确性。

```
# 文件：python/sglang/srt/layers/attention/triton_backend.py
# 修复 1：_update_draft_extend_buffers 中 qo_indptr stride

def _update_draft_extend_buffers(self, bs, seq_lens, req_pool_indices, forward_mode, spec_info)
:
    seq_lens = seq_lens[:bs]
    # V2 draft-extend 每请求填充 num_draft_tokens 个 token（与 CUDA Graph runner
    布局一致）；
    # num_steps+1 仅当 topk==1 时等于 num_draft_tokens。
    num_tokens_per_bs = (
        self.num_draft_tokens
        if forward_mode.is_draft_extend_v2()
        else self.speculative_num_steps + 1
    )
    qo_indptr = self.qo_indptr[: bs + 1]
    qo_indptr[: bs + 1] = torch.arange(
        0,
        bs * num_tokens_per_bs + 1,
        step=num_tokens_per_bs,
        dtype=torch.int32,
        device=self.device,
    )
    # … 后续 kv_indptr 逻辑不变 …

# 修复 2：_build_cuda_graph_forward_metadata 中 max_extend_len
# draft_extend 分支（include_v2=True）
elif forward_mode.is_draft_extend(include_v2=True):
    return ForwardMetadata(
        attn_logits=None,
        attn_lse=None,
        # 必须与用于构建 qo_indptr 的每请求 query 数量（num_tokens_per_bs）一致；
        # 否则 topk>1 时 num_draft_tokens > num_steps+1，kernel grid 过小导致丢 block。
        max_extend_len=(
            self.num_draft_tokens
            if forward_mode.is_draft_extend_v2()
            else self.speculative_num_steps + 1
        ),
        num_kv_splits=None,
        kv_indptr=self.kv_indptr[: bs + 1],
        kv_indices=self.cuda_graph_kv_indices,
        qo_indptr=self.qo_indptr[: bs + 1],
        custom_mask=None,
        mask_indptr=None,
        # … 其余参数不变 …
    )
```

评论区精华

gemini-code-assist[bot] 在 review 时指出第一处修复不完整：`_build_cuda_graph_forward_metadata` 中的 `max_extend_len` 仍使用 `speculative_num_steps + 1`，会导致 kernel grid 过小、丢 block。提交者随后在第二个 commit 中补充了该修复，已采纳建议。

- max_extend_len 未同步更新 (correctness): 提交者在第二个 commit 中采纳建议，将 max_extend_len 改为条件分支使用 num_draft_tokens。

风险与影响

- 风险：本 PR 修改集中在 CUDA Graph 元数据构建两个函数，影响范围仅限于 EAGLE spec-v2 的 draft-extend 模式。变更很小，且逻辑正确性经过测试验证。核心回归风险是确保其他 forward mode（如 target_verify、非 V2 draft-extend）不受影响——条件分支确保了这一点。未观察到性能或兼容性风险。
- 影响：影响范围限定在使用 EAGLE spec-v2、CUDA Graph、topk>1 且 bs>1 的推理场景。修复后消除了该场景下的 NaN 错误，使 Triton 后端在组合使用以上特性时行为正确。对用户可见的错误可直接消除。
- 风险标记：CUDA Graph 元数据不一致

关联脉络

- PR #26651 Fix fill_len asymmetric assignment statement in ignore-eos branch: PR body 和注释中提及此 PR 作为 capture-time warmup 退避策略的参考，表明本修复与之前调度器 fill_len 修复有关联。
- PR #27486 [spec] Misc defensive guards for EAGLE draft KV indexing: 同为针对 EAGLE 投机解码的修复，涉及相似的 CUDA Graph 和 KV 索引路径，与该 PR 属于同一功能线。